# Bootstrapping Deep Lexical Resources: Resources for Courses

**Timothy Baldwin**
Department of Computer Science and Software Engineering
University of Melbourne, Victoria 3010 Australia
tim@csse.unimelb.edu.au

## Abstract

We propose a range of deep lexical acquisition methods which make use of morphological, syntactic and ontological language resources to model word similarity and bootstrap from a seed lexicon. The different methods are deployed in learning lexical items for a precision grammar, and shown to each have strengths and weaknesses over different word classes. A particular focus of this paper is the relative accessibility of different language resource types, and predicted "bang for the buck" associated with each in deep lexical acquisition applications.

## 1 Introduction

Over recent years, computational linguistics has benefitted considerably from advances in statistical modelling and machine learning, culminating in methods capable of deeper, more accurate automatic analysis, over a wider range of languages. Implicit in much of this work, however, has been the existence of **deep language resources** (DLR hereafter) of ever-increasing linguistic complexity, including lexical semantic resources (e.g. WordNet and FrameNet), precision grammars (e.g. the English Resource Grammar and the various ParGram grammars) and richly-annotated treebanks (e.g. PropBank and CCGbank).

Due to their linguistic complexity, DLRs are invariably constructed by hand and thus restricted in size and coverage. Our aim in this paper is to develop general-purpose automatic methods which can be used to automatically expand the coverage of an existing DLR, through the process of **deep lexical acquisition** (DLA hereafter).

The development of DLRs can be broken down into two basic tasks: (1) design of a data representation to systematically capture the generalisations and idiosyncracies of the dataset of interest (**system design**); and (2) classification of data items according to the predefined data representation (**data classification**). In the case of a deep grammar, for example, system design encompasses the construction of the system of lexical types, templates, and/or phrase structure rules, and data classification corresponds to the determination of the lexical type(s) each individual lexeme conforms to. DLA pertains to the second of these tasks, in automatically mapping a given lexeme onto a pre-existing system of lexical types associated with a DLR.

We propose to carry out DLA through a bootstrap process, that is by employing some notion of word similarity, and learning the lexical types for a novel lexeme through analogy with maximally similar word(s) for which we know the lexical types. In this, we are interested in exploring the impact of different secondary language resources (LRs) on DLA, and estimating how successfully we can expect to learn new lexical items from a range of LR types. That is, we estimate the expected DLA "bang for the buck" from a range of secondary LR types of varying size and complexity. As part of this, we look at the relative impact of different LRs on DLA for different open word classes, namely nouns, verbs, adjectives and adverbs.

We demonstrate the proposed DLA methods relative to the English Resource Grammar (see Section 2.1), and in doing so assume the lexical types of the target DLR to be syntactico-semantic in nature. For example, we may predict that the word *dog* has a usage as an intransitive countable noun (`n_intr_le`,[1] cf. *The dog barked*), and also as a transitive verb (`v_np_trans_le`, cf. *It dogged my every step*).

A secondary interest of this paper is the consideration of how well we could expect to perform DLA for languages of differing density, from "low-

[1] All example lexical types given in this paper are taken directly from the English Resource Grammar – see Section 2.1.

density" languages (such as Walpiri or Uighur) for which we have limited LRs, to "high-density" languages (such as English or Japanese) for which we have a wide variety of LRs. To this end, while we exclusively target English in this paper, we experiment with a range of LRs of varying complexity and type, including morphological, syntactic and ontological LRs. Note that we attempt to maintain consistency across the feature sets associated with each, to make evaluation as equitable as possible.

The remainder of this paper is structured as follows. Section 2 outlines the process of DLA and reviews relevant resources and literature. Sections 3, 4 and 5 propose a range of DLA methods based on morphology, syntax and ontological semantics, respectively. Section 6 evaluates the proposed methods relative to the English Resource Grammar.

## 2 Task Outline

This research aims to develop methods for DLA which can be run automatically given: (a) a pre-existing DLR which we wish to expand the coverage of, and (b) a set of secondary LRs/preprocessors for that language. The basic requirements to achieve this are the discrete inventory of lexical types in the DLR, and a pre-classification of each secondary LR (e.g. as a corpus or wordnet, to determine what set of features to employ). Beyond this, we avoid making any assumptions about the language family or DLR type.

The DLA strategy we propose in this research is to use secondary LR(s) to arrive at a feature signature for each lexeme, and map this onto the system of choice indirectly via supervised learning, i.e. observation of the correlation between the feature signature and classification of bootstrap data. This methodology can be applied to unannotated corpus data, for example, making it possible to tune a lexicon to a particular domain or register as exemplified in a particular repository of text. As it does not make any assumptions about the nature of the system of lexical types, we can apply it fully automatically to any DLR and feed the output directly into the lexicon without manual intervention or worry of misalignment. This is a distinct advantage when the inventory of lexical types is continually undergoing refinement, as is the case with the English Resource Grammar (see below).

A key point of interest in this paper is the investigation of the relative "bang for the buck" when different types of LR are used for DLA. Crucially, we investigate only LRs which we believe to be plausibly available for languages of varying density, and aim to minimise assumptions as to the pre-existence of particular preprocessing tools. The basic types of resources and tools we experiment with in this paper are detailed in Table 1.

Past research on DLA falls into two basic categories: expert system-style DLA customised to learning particular linguistic properties, and DLA via resource translation. In the first instance, a specialised methodology is proposed to (automatically) learn a particular linguistic property such as verb subcategorisation (e.g. Korhonen (2002)) or noun countability (e.g. Baldwin and Bond (2003a)), and little consideration is given to the applicability of that method to more general linguistic properties. In the second instance, we take one DLR and map it onto another to arrive at the lexical information in the desired format. This can take the form of a one-step process, in mining lexical items directly from a DLR (e.g. a machine-readable dictionary (Sanfilippo and Poznański, 1992)), or two-step process in reusing an existing system to learn lexical properties in one format and then mapping this onto the DLR of choice (e.g. Carroll and Fang (2004) for verb subcategorisation learning).

There have also been instances of more general methods for DLA, aligned more closely with this research. Fouvry (2003) proposed a method of token-based DLA for unification-based precision grammars, whereby partially-specified lexical features generated via the constraints of syntactically-interacting words in a given sentence context, are combined to form a consolidated lexical entry for that word. That is, rather than relying on indirect feature signatures to perform lexical acquisition, the DLR itself drives the incremental learning process. Also somewhat related to this research is the general-purpose verb feature set proposed by Joanis and Stevenson (2003), which is shown to be applicable in a range of DLA tasks relating to English verbs.

### 2.1 English Resource Grammar

All experiments in this paper are targeted at the **English Resource Grammar** (ERG; Flickinger (2002), Copestake and Flickinger (2000)). The ERG is an implemented open-source broad-coverage pre-

| ***Secondary LR type*** | ***Description*** | ***Preprocessor(s)*** |
|---|---|---|
| Word list$^{***}$ | List of words with basic POS | — |
| Morphological lexicon$^{*}$ | Derivational and inflectional word relations | — |
| Compiled corpus$^{***}$ | Unannotated text corpus | POS tagger$^{**}$<br>Chunk parser$^{*}$<br>Dependency parser$^{*}$ |
| WordNet-style ontology$^{*}$ | Lexical semantic word linkages | — |

Table 1: Secondary LR and tool types targeted in this research ($^{***}$ = high expectation of availability for a given language; $^{**}$ = medium expectation of availability; $^{*}$ = low expectation of availability)

cision Head-driven Phrase Structure Grammar (HPSG) developed for both parsing and generation. It contains roughly 10,500 lexical items, which, when combined with 59 lexical rules, compile out to around 20,500 distinct word forms.[2] Each lexical item consists of a unique identifier, a lexical type (one of roughly 600 leaf types organized into a type hierarchy with a total of around 4,000 types), an orthography, and a semantic relation. The grammar also contains 77 phrase structure rules which serve to combine words and phrases into larger constituents. Of the 10,500 lexical items, roughly 3,000 are multiword expressions.

To get a basic sense of the syntactico-semantic granularity of the ERG, the noun hierarchy, for example, is essentially a cross-classification of countability/determiner co-occurrence, noun valence and preposition selection properties. For example, lexical entries of `n_mass_count_ppof_le` type can be either countable or uncountable, and optionally select for a PP headed by *of* (example lexical items are *choice* and *administration*).

As our target lexical type inventory for DLA, we identified all open-class lexical types with at least 10 lexical entries, under the assumption that: (a) the ERG has near-complete coverage of closed-class lexical entries, and (b) the bulk of new lexical entries will correspond to higher-frequency lexical types. This resulted in the following breakdown:[3]

| ***Word class*** | ***Lexical types*** | ***Lexical items*** |
|---|---|---|
| Noun | 28 | 3,032 |
| Verb | 39 | 1,334 |
| Adjective | 17 | 1,448 |
| Adverb | 26 | 721 |
| Total | 110 | 5,675 |

Note that it is relatively common for a lexeme to occur with more than one lexical type in the ERG: 22.6% of lexemes have more than one lexical type, and the average number of lexical types per lexeme is 1.12.

In evaluation, we assume we have prior knowledge of the basic word classes each lexeme belongs to (i.e. noun, verb, adjective and/or adverb), information which could be derived trivially from preexisting shallow lexicons and/or the output of a tagger.

Recent development of the ERG has been tightly coupled with treebank annotation, and all major versions of the grammar are deployed over a common set of treebank data to help empirically trace the evolution of the grammar and retrain parse selection models (Oepen et al., 2002). We treat this as a held-out dataset for use in analysis of the *token* frequency of each lexical item, to complement analysis of *type*-level learning performance (see Section 6).

### 2.2 Classifier design

The proposed procedure for DLA is to generate a feature signature for each word contained in a given secondary LR, take the subset of lexemes contained in the original DLR as training data, and learn lexical items for the remainder of the lexemes through supervised learning. In order to maximise comparability between the results for the different DLRs, we employ a common classifier design wherever possible (in all cases other than ontology-based

[2] All statistics and analysis relating to the ERG in this paper are based on the version of 11 June, 2004.

[3] Note that all results are over simplex lexemes only, and that we choose to ignore multiword expressions in this research.

DLA), using TiMBL 5.0 (Daelemans et al., 2003); we used the IB1 $k$-NN learner implementation within TiMBL, with $k$ = 9 throughout.[4] We additionally employ the feature selection method of Baldwin and Bond (2003b), which generates a combined ranking of all features in descending order of "informativeness" and skims off the top-$N$ features for use in classification; $N$ was set to 100 in all experiments.

As observed above, a significant number of lexemes in the ERG occur in multiple lexical items. If we were to take all lexical type combinations observed for a single lexeme, the total number of lexical "super"-types would be 451, of which 284 are singleton classes. Based on the sparseness of this data and also the findings of Baldwin and Bond (2003b) over a countability learning task, we choose to carry out DLA via a suite of 110 binary classifiers, one for each lexical type.

We deliberately avoid carrying out extensive feature engineering over a given secondary LR, choosing instead to take a varied but simplistic set of features which is parallelled as much as possible between LRs (see Sections 3–5 for details). We additionally tightly constrain the feature space to a maximum of 3,900 features, and a maximum of 50 feature instances for each feature type; in each case, the 50 feature instances are selected by taking the features with highest saturation (i.e. the highest ratio of non-zero values) across the full lexicon. This is in an attempt to make evaluation across the different secondary LRs as equitable as possible, and get a sense of the intrinsic potential of each secondary LR in DLA. Each feature instance is further translated into two feature values: the raw count of the feature instance for the target word in question, and the relative occurrence of the feature instance over all target word token instances.

One potential shortcoming of our classifier architecture is that a given word can be negatively classified by all unit binary classifiers and thus not assigned any lexical items. In this case, we fall back on the majority-class lexical type for each word class the word has been pre-identified as belonging to.

## 3 Morphology-based Deep Lexical Acquisition

We first perform DLA based on the following morphological LRs: (1) word lists, and (2) morphological lexicons with a description of derivational word correspondences. Note that in evaluation, we presuppose that we have access to word lemmas although in the first instance, it would be equally possible to run the method over non-lemmatised data.[5]

### 3.1 Character $n$-grams

In line with our desire to produce DLA methods which can be deployed over both low- and high-density languages, our first feature representation takes a simple word list and converts each lexeme into a character $n$-gram representation.[6] In the case of English, we generated all 1- to 6-grams for each lexeme, and applied a series of filters to: (1) filter out all $n$-grams which occurred less than 3 times in the lexicon data; and (2) filter out all $n$-grams which occur with the same frequency as larger $n$-grams they are proper substrings of. We then select the 3,900 character $n$-grams with highest saturation across the lexicon data (see Section 2.2).

The character $n$-gram-based classifier is the simplest of all classifiers employed in this research, and can be deployed on any language for which we have a word list (ideally lemmatised).

### 3.2 Derviational morphology

The second morphology-based DLA method makes use of derivational morphology and analysis of the process of word formation. As an example of how derivational information could assist DLA, knowing that the noun *achievement* is deverbal and incorporates the *-ment* suffix is a strong predictor of it being optionally uncountable and optionally selecting for a PP argument (i.e. being of lexical type `n_mass_count_ppof_le`).

We generate derivational morphological features for a given lexeme by determining its word cluster in CATVAR[7] (Habash and Dorr, 2003) and then for each sister lexeme (i.e. lexeme occurring in the

[4] We also experimented with bsvm and SVMLight, and a maxent toolkit, but found TiMBL to be superior overall, we hypothesise due to the tight integration of continuous features in TiMBL.

[5] Although this would inevitably lose lexical generalisations among the different word forms of a given lemma.

[6] We also experimented with syllabification, but found the character $n$-grams to produce superior results.

[7] In the case that the a given lemma is not in CATVAR, we attempt to dehyphenate and then deprefix the word to find a match, failing which we look for the lexeme of smallest edit distance.

same cluster as the original lexeme with the same word stem), determine if there is a series of edit operations over suffixes and prefixes which maps the lexemes onto one another. For each sister lexeme where such a correspondence is found to exist, we output the nature of the character transformation and the word classes of the lexemes involved. E.g., the sister lexemes for *achievement*$_{\mathsf{N}}$ in CATVAR are *achieve*$_{\mathsf{V}}$, *achiever*$_{\mathsf{N}}$, *achievable*$_{\mathsf{Adj}}$ and *achievability*$_{\mathsf{N}}$; the mapping between *achievement*$_{\mathsf{N}}$ and *achiever*$_{\mathsf{N}}$, e.g., would be analysed as:

$$\texttt{N} -\texttt{ment\$} \rightarrow \texttt{N} +\texttt{r\$}$$

Each such transformation is treated as a single feature.

We exhaustively generate all such transformations for each lexeme, and filter the feature space as for character $n$-grams above.

Clearly, LRs which document derivational morphology are typically only available for high-density languages. Also, it is worth bearing in mind that derivational morphology exists in only a limited form for certain language families, e.g. agglutinative languages.

## 4 Syntax-based Deep Lexical Acquisition

Syntax-based DLA takes a raw text corpus and preprocesses it with either a tagger, chunker or dependency parser. It then extracts a set of 39 feature types based on analysis of the token occurrences of a given lexeme, and filters over each feature type to produce a maximum of 50 feature instances of highest saturation (e.g. if the feature type is the word immediately proceeding the target word, the feature instances are the 50 words which proceed the most words in our lexicon). The feature signature associated with a word for a given preprocessor type will thus have a maximum of 3,900 items ($39 \times 50 \times 2$).[8]

### 4.1 Tagging

The first and most basic form of syntactic preprocessing is part-of-speech (POS) tagging. For our purposes, we use a Penn treebank-style tagger custom-built using fnTBL 1.0 (Ngai and Florian, 2001), and further lemmatise the output of the tagger using morph (Minnen et al., 2000).

The feature types used with the tagger are detailed in Table 2, where the position indices are relative to the target word (e.g. the word at position $-2$ is two words to the left of the target word, and the POS tag at position $0$ is the POS of the target word). All features are relative to the POS tags and words in the immediate context of each token occurrence of the target word. “Bi-words” are word bigrams (e.g. biword $(1, 3)$ is the bigram made up of the words one and three positions to the right of the target word); “bi-tags” are, similarly, POS tag bigrams.

### 4.2 Chunking

The second form of syntactic preprocessing, which builds directly on the output of the POS tagger, is CoNLL 2000-style full text chunking (Tjong Kim Sang and Buchholz, 2000). The particular chunker we use was custom-built using fnTBL 1.0 once again, and operates over the lemmatised output of the POS tagger.

The feature set for the chunker output includes a subset of the POS tagger features, but also makes use of the local syntactic structure in the chunker input in incorporating both intra-chunk features (such as modifiers of the target word if it is the head of a chunk, or the head if it is a modifier) and inter-chunk features (such as surrounding chunk types when the target word is chunk head). See Table 2 for full details.

Note that while chunk parsers are theoretically easier to develop than full phrase-structure or treebank parsers, only high-density languages such as English and Japanese have publicly available chunk parsers.

### 4.3 Dependency parsing

The third and final form of syntactic preprocessing is dependency parsing, which represents the pinnacle of both robust syntactic sophistication and inaccessibility for any other than the highest-density languages.

The particular dependency parser we use is RASP[9] (Briscoe and Carroll, 2002), which outputs head–modifier dependency tuples and further classifies each tuple according to a total of 14 relations; RASP also outputs the POS tag of each word token. As our features, we use both local word and POS features, for comparability with the POS tagger

[8] Note that we will have less than 50 feature instances for some feature types, e.g. the POS tag of the target word, given that the combined size of the Penn POS tagset is 36 elements (not including punctuation).

[9] RASP is, strictly speaking, a full syntactic parser, but we use it in dependency parser mode

| ***Feature type*** | ***Positions/description*** | ***Total*** |
|---|---|---|
| ***TAGGER*** | | ***39*** |
| POS tag | $(-4, -3, -2, -1, 0, 1, 2, 3, 4)$ | 9 |
| Word | $(-4, -3, -2, -1, 1, 2, 3, 4)$ | 8 |
| POS bi-tag | $(\,(-4, -1), (-4, 0), (-3, -2), (-3, -1), (-3, 0), (-2, -1), (-2, 0),$ $(-1, 0), (0, 1), (0, 2), (0, 3), (0, 4), (1, 2), (1, 3), (1, 4), (2, 3)\,)$ | 16 |
| Bi-word | $((-3, -2), (-3, -1), (-2, -1), (1, 2), (1, 3), (2, 3))$ | 6 |
| ***CHUNKER*** | | ***39*** |
| Modifier$_{\text{head}}$ | Chunk heads when target word is modifier | 1 |
| Modifier$_{\text{chunk}}$ | Chunk types when target word is modifier | 1 |
| Modifiee$_{\text{word}}$ | Modifiers when target word is chunk head | 1 |
| Modifiee$_{\text{POS}}$ | POS tag of modifiers when target word is chunk head | 1 |
| Modifiee$_{\text{word+POS}}$ | Word + POS tag of modifiers when target word is chunk head | 1 |
| POS tag | $(-3, -2, -1, 0, 1, 2, 3)$ | 7 |
| Word | $(-3, -2, -1, 1, 2, 3)$ | 6 |
| Chunk | $(-4, -3, -2, -1, 0, 1, 2, 3, 4)$ | 9 |
| Chunk head | $(-3, -2, -1, 1, 2, 3)$ | 6 |
| Bi-chunk | $((-2, -1), (-2, 0), (-1, 0), (0, 1), (0, 2), (1, 2))$ | 6 |
| ***DEPENDENCY PARSER*** | | ***39*** |
| POS tag | $(-2, -1, 0, 1, 2)$ | 5 |
| Word | $(-2, -1, 1, 2)$ | 4 |
| Conj$_{\text{word}}$ | Words the target word coordinates with | 1 |
| Conj$_{\text{POS}}$ | POS of words the target word coordinates with | 1 |
| Head | Head word when target word modifier in dependency relation (× 14) | 14 |
| Modifier | Modifier when target word head of dependency relation (× 14) | 14 |

Table 2: Feature types used in syntax-based DLA for the different preprocessors

and chunker, and also dependency-derived features, namely the modifier of all dependency tuples the target word occurs as head of, and conversely, the head of all dependency tuples the target word occurs as modifier in, along with the dependency relation in each case. See Table 2 for full details.

### 4.4 Corpora

We ran the three syntactic preprocessors over a total of three corpora, of varying size: the Brown corpus (∼460K tokens) and Wall Street Journal corpus (∼1.2M tokens), both derived from the Penn Treebank (Marcus et al., 1993), and the written component of the British National Corpus (∼98M tokens: Burnard (2000)). This selection is intended to model the effects of variation in corpus size, to investigate how well we could expect syntax-based DLA methods to perform over both smaller and larger corpora.

Note that the only corpus annotation we make use of is sentence tokenisation, and that all preprocessors are run automatically over the raw corpus data. This is in an attempt to make the methods maximally applicable to lower-density languages where annotated corpora tend not to exist but there is at least the possibility of accessing raw text collections.

## 5 Ontology-based Deep Lexical Acquisition

The final DLA method we explore is based on the hypothesis that there is a strong correlation between the semantic and syntactic similarity of words, a claim which is best exemplified in the work of Levin (1993) on diathesis alternations. In our case, we take word similarity as given and learn the syntactic behaviour of novel words relative to semantically-similar words for which we know the lexical types. We use WordNet 2.0 (Fellbaum, 1998) to determine word similarity, and for each sense of the target word in WordNet: (1) construct the set of “semantic neighbours” of that word sense, comprised of all synonyms, direct hyponyms and direct hypernyms; and (2) take a majority vote across the lexical types of the semantic neighbours which occur in the training data. Note that this diverges from the learning paradigm adopted for the morphology- and syntax-based DLA methods in that we use a simple voting strategy rather than relying on an external learner to carry out the classification. The full set of lexical entries for the target word is generated by taking the union of the majority votes across all senses of the word, such that a polysemous lexeme can potentially give rise to multiple lexical entries.

This learning procedure is based on the method used by van der Beek and Baldwin (2004) to learn Dutch countability.

As for the suite of binary classifiers, we fall back on the majority class lexical type as the default in the instance that a given lexeme is not contained in WordNet 2.0 or no classification emerges from the set of semantic neighbours. It is important to realise that WordNet-style ontologies exist only for the highest-density languages, and that this method will thus have very limited language applicability.

## 6 Evaluation

We evaluate the component methods over the 5,675 open-class lexical items of the ERG described in Section 2.1 using 10-fold stratified cross-validation. In each case, we calculate the **type precision** (the proportion of correct hypothesised lexical entries) and **type recall** (the proportion of gold-standard lexical entries for which we get a correct hit), which we roll together into the **type F-score** (the harmonic mean of the two) relative to the gold-standard ERG lexicon. We also measure the **token accuracy** for the lexicon derived from each method, relative to the Redwoods treebank of Verbmobil data associated with the ERG (see Section 2.1).[10] The token accuracy represents a weighted version of type precision, relative to the distribution of each lexical item in a representative text sample, and provides a crude approximation of the impact of each DLA method on parser coverage. That is, it gives more credit for a method having correctly hypothesised a commonly-occurring lexical item than a low-frequency lexical item, and no credit for having correctly identified a lexical item not occurring in the corpus.

The overall results are presented in Figure 1, which are then broken down into the four open word classes in Figures 2–5. The baseline method (*Base*) in each case is a simple majority-class classifier, which generates a unique lexical item for each lexeme pre-identified as belonging to a given word class of the following type:

| Word class | Majority-class lexical type |
|---|---|
| Noun | n_intr_le |
| Verb | v_np_trans_le |
| Adjective | adj_intrans_le |
| Adverb | adv_int_vp_le |

In each graph, we present the type F-score and token accuracy for each method, and mark the best-performing method in terms of each of these evaluation measures with a star ($\star$). The results for syntax-based DLA ($S_{\text{POS}}$, $S_{\text{CHUNK}}$ and $S_{\text{PARSE}}$) are based on the BNC in each case. We return to investigate the impact of corpus size on the performance of the syntax-based methods below.

Looking first at the combined results over all lexical types (Figure 1), the most successful method in terms of type F-score is syntax-based DLA, with chunker-based preprocessing marginally outperforming tagger- and parser-based preprocessing (type F-score = 0.641). The most successful method in terms of token accuracy is ontology-based DLA (token accuracy = 0.544).

The figures for token accuracy require some qualification: ontology-based DLA tends to be liberal in its generation of lexical items, giving rise to over 20% more lexical items than the other methods (7,307 vs. 5-6000 for the other methods) and proportionately low type precision. This correlates with an inherent advantage in terms of token accuracy, which we have no way of balancing up in our token-based evaluation, as the treebank data offers no insight into the true worth of false negative lexical items (i.e. have no way of distinguishing between unobserved lexical items which are plain wrong from those which are intuitively correct and could be expected to occur in alternate sets of treebank data). We leave investigation of the impact of these extra lexical items on the overall parser performance (in terms of chart complexity and parse selection) as an item for future research.

The morphology-based DLA methods were around baseline performance overall, with character $n$-grams marginally more successful than derivational morphology in terms of both type F-score and token accuracy.

Turning next to the results for the proposed methods over nouns, verbs, adjectives and adverbs (Figures 2–5, respectively), we observe some interesting effects. First, morphology-based DLA hovers around baseline performance for all word classes except adjectives, where character $n$-grams produce the highest F-score of all methods, and nouns, where derivational morphology seems to aid DLA slightly (providing weak support for our original hypothesis in Section 3.2 relating to deverbal nouns and affixation).

[10] Note that the token accuracy is calculated only over the open-class lexical items, not the full ERG lexicon.

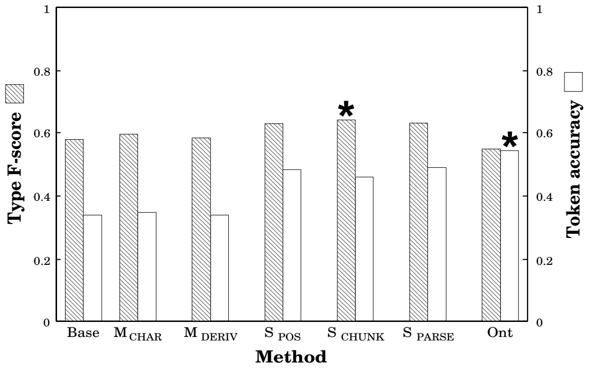


Figure 1: Results for the proposed deep lexical acquisition methods over ALL lexical types

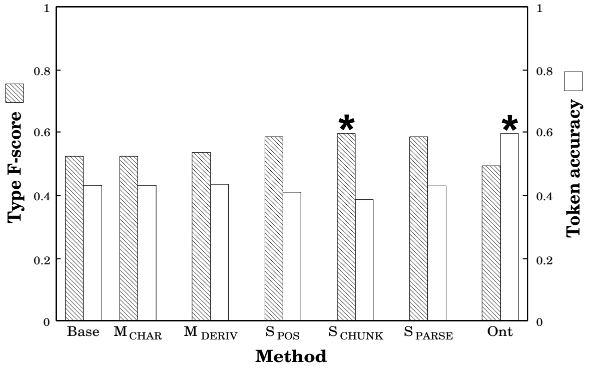


Figure 2: Results for the proposed deep lexical acquisition methods over NOUN lexical types

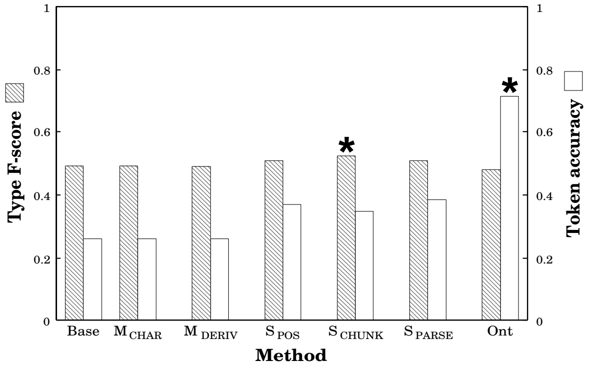


Figure 3: Results for the proposed deep lexical acquisition methods over VERB lexical types

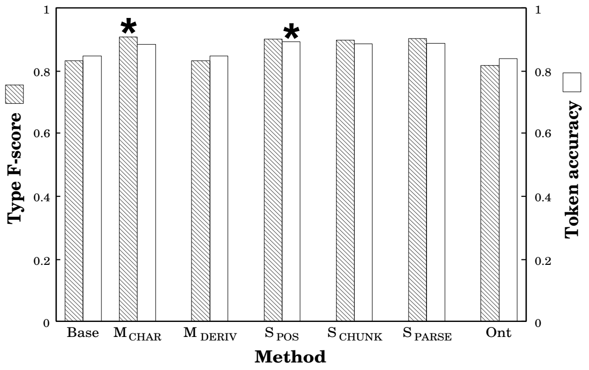


Figure 4: Results for the proposed deep lexical acquisition methods over ADJECTIVE lexical types

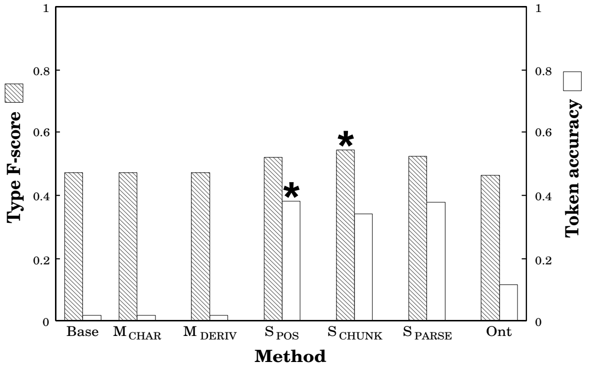


Figure 5: Results for the proposed deep lexical acquisition methods over ADVERB lexical types

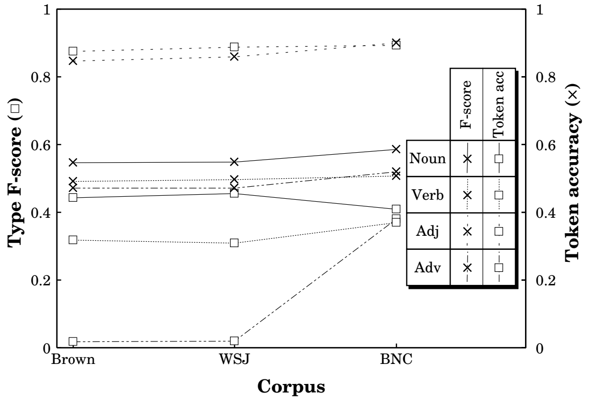


Figure 6: Results for the syntax-based deep lexical acquisition methods over corpora of differing size

*Note:* Base = baseline, $\mathrm{M}_{\mathrm{CHAR}}$ = morphology-based DLA with character $n$-grams, $\mathrm{M}_{\mathrm{DERIV}}$ = derivational morphology-based DLA, $\mathrm{S}_{\mathrm{POS}}$ = syntax-based DLA with POS tagging, $\mathrm{S}_{\mathrm{CHUNK}}$ = syntax-based DLA with chunking, $\mathrm{S}_{\mathrm{PARSE}}$ = syntax-based DLA with dependency parsing, and Ont = ontology-based DLA

Syntax-based DLA leads to the highest type F-score for nouns, verbs and adverbs, and the highest token accuracy for adjectives and adverbs. The differential in results between syntax-based DLA and the other methods is particularly striking for adverbs, with a maximum type F-score of 0.544 (for chunker-based preprocessing) and token accuracy of 0.340 (for tagger-based preprocessing), as compared to baseline figures of 0.471 and 0.017 respectively. There is relatively little separating the three styles of preprocessing in syntax-based DLA, although chunker-based preprocessing tends to have a slight edge in terms of type F-score, and tagger-based preprocessing generally produces the highest token accuracy.[11] This suggests that access to a POS tagger for a given language is sufficient to make syntax-based DLA work, and that syntax-based DLA thus has moderately high applicability across languages of different densities.

Ontology-based DLA is below baseline in terms of type F-score for all word classes, but results in the highest token accuracy of all methods for nouns and verbs (although this finding must be taken with a grain of salt, as noted above).

Another noteworthy feature of Figures 2–5 is the huge variation in absolute performance across the word classes: adjectives are very predictable, with a majority class-based baseline type F-score of 0.832 and token accuracy of 0.847; adverbs, on the other hand, are similar to verbs and nouns in terms of their baseline type F-score (at 0.471), but the adverbs that occur commonly in corpus data appear to belong to less-populated lexical types (as seen in the baseline token accuracy of a miniscule 0.017). Nouns appear the hardest to learn in terms of the relative increment in token accuracy over the baseline. Verbs are extremely difficult to get right at the type level, but it appears that ontology-based DLA is highly adept at getting the commonly-occurring lexical items right.

To summarise these findings, adverbs seem to benefit the most from syntax-based DLA. Adjectives, on the other hand, can be learned most effectively from simple character $n$-grams, i.e. similarly-spelled adjectives tend to have similar syntax, a somewhat surprising finding. Nouns are surprisingly hard to learn, but seem to benefit to some degree from corpus data and also ontological similarity. Lastly, verbs pose a challenge to all methods at the type level, but ontology-based DLA seems to be able to correctly predict the commonly-occurring lexical entries.

Finally, we examine the impact of corpus size on the performance of syntax-based DLA with tagger-based preprocessing.[12] In Figure 6, we examine the relative change in type F-score and token accuracy across the four word classes as we increase the corpus size (from 0.5m words to 1m and finally 100m words, in the form of the Brown corpus, WSJ corpus and BNC, respectively). For verbs and adjectives, there is almost no change in either type F-score or token accuracy when we increase the corpus size, whereas for nouns, the token accuracy actually drops slightly. For adverbs, on the other hand, the token accuracy jumps up from 0.020 to 0.381 when we increase the corpus size from 1m words to 100m words, while the type F-score rises only slightly. It thus seems to be the case that large corpora have a considerable impact on DLA for commonly-occurring adverbs, but that for the remaining word classes, it makes little difference whether we have 0.5m or 100m words. This can be interpreted either as evidence that modestly-sized corpora are good enough to perform syntax-based DLA over (which would be excellent news for low-density languages!), or alternatively that for the simplistic syntax-based DLA methods proposed here, more corpus data is not the solution to achieving higher performance.

Returning to our original question of the "bang for the buck" associated with individual LRs, there seems to be no simple answer: simple word lists are useful in learning the syntax of adjectives in particular, but offer little in terms of learning the other three word classes. Morphological lexicons with derivational information are moderately advantageous in learning the syntax of nouns but little else. A POS tagger seems sufficient to carry out syntax-based DLA, and the word class which benefits the most from larger amounts of corpus data is adverbs, otherwise the proposed syntax-based DLA methods don't seem to benefit from larger-sized corpora. Ontologies have the greatest impact on verbs and, to a lesser degree, nouns. Ultimately, this seems to lend weight to a "horses for courses", or perhaps "resources for courses" approach to DLA.

[11] This trend was observed across all three corpora, although we do no present the full results here.

[12] The results for chunker- and parser-based preprocessing are almost identical, and this omitted from the paper.

## 7 Conclusion

We have proposed three basic paradigms for deep lexical acquisition, based on morphological, syntactic and ontological language resources, and demonstrated the effectiveness of each strategy at learning lexical items for the lexicon of a precision English grammar. We discovered surprising variation in the results for the different DLA methods, with each learning method performing particularly well for at least one basic word class, but the best overall methods being syntax- and ontology-based DLA.

The results presented in this paper are based on one particular language (English) and a very specific style of DLR (a precision grammar, namely the English Resource Grammar), so some caution must be exercised in extrapolating the results too liberally over new languages/DLA tasks. In future research, we are interested in carrying out experiments over other languages and alternate DLRs to determine how well these results generalise and formulate alternate strategies for DLA.

### Acknowledgements

This material is based upon work supported in part by NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation. We would like to thank the members of the University of Melbourne LT group and the three anonymous reviewers for their valuable input on this research.

## References

[Baldwin and Bond2003a] Timothy Baldwin and Francis Bond. 2003a. Learning the countability of English nouns from corpus data. In *Proc. of the 41st Annual Meeting of the ACL*, pages 463–70, Sapporo, Japan.

[Baldwin and Bond2003b] Timothy Baldwin and Francis Bond. 2003b. A plethora of methods for learning English countability. In *Proc. of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP 2003)*, pages 73–80, Sapporo, Japan.

[Briscoe and Carroll2002] Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proc. of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1499–1504, Las Palmas, Canary Islands.

[Burnard2000] Lou Burnard. 2000. *User Reference Guide for the British National Corpus*. Technical report, Oxford University Computing Services.

[Carroll and Fang2004] John Carroll and Alex Fang. 2004. The automatic acquisition of verb subcategorisations and their impact on the performance of an HPSG parser. In *Proc. of the First International Joint Conference on Natural Language Processing (IJCNLP-04)*, pages 107–14, Sanya City, China.

[Copestake and Flickinger2000] Ann Copestake and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proc. of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, Athens, Greece.

[Daelemans et al.2003] Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2003. *TiMBL: Tilburg Memory Based Learner, version 5.0, Reference Guide*. ILK Technical Report 03-10.

[Fellbaum1998] Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, USA.

[Flickinger2002] Dan Flickinger. 2002. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering*. CSLI Publications, Stanford, USA.

[Fouvry2003] Frederik Fouvry. 2003. *Robust Processing for Constraint-based Grammar Formalisms*. Ph.D. thesis, University of Essex.

[Habash and Dorr2003] Nizar Habash and Bonnie Dorr. 2003. CATVAR: A database of categorial variations for English. In *Proc. of the Ninth Machine Translation Summit (MT Summit IX)*, pages 471–4, New Orleans, USA.

[Joanis and Stevenson2003] Eric Joanis and Suzanne Stevenson. 2003. A general feature space for automatic verb classification. In *Proc. of the 10th Conference of the EACL (EACL 2003)*, pages 163–70, Budapest, Hungary.

[Korhonen2002] Anna Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, University of Cambridge.

[Levin1993] Beth Levin. 1993. *English Verb Classes and Alterations*. University of Chicago Press, Chicago, USA.

[Marcus et al.1993] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–30.

[Minnen et al.2000] Guido Minnen, John Carroll, and Darren Pearce. 2000. Robust, applied morphological generation. In *Proceedings of the first International Natural Language Genration Conference*, Mitzpe Ramon, Israel.

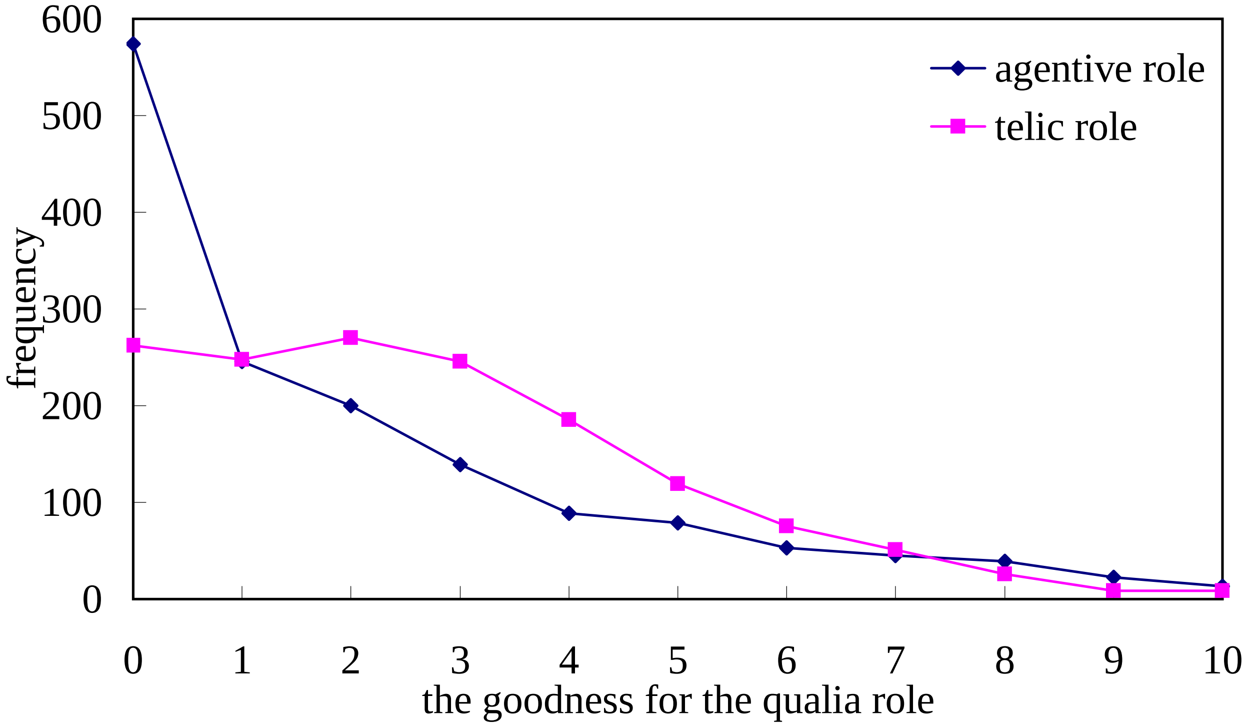
[Ngai and Florian2001] Grace Ngai and Radu Florian. 2001. Transformation-based learning in the fast lane. In *Proc. of the 2nd Annual Meeting of the North American Chapter of Association for Computational Linguistics (NAACL2001)*, pages 40–7, Pittsburgh, USA.

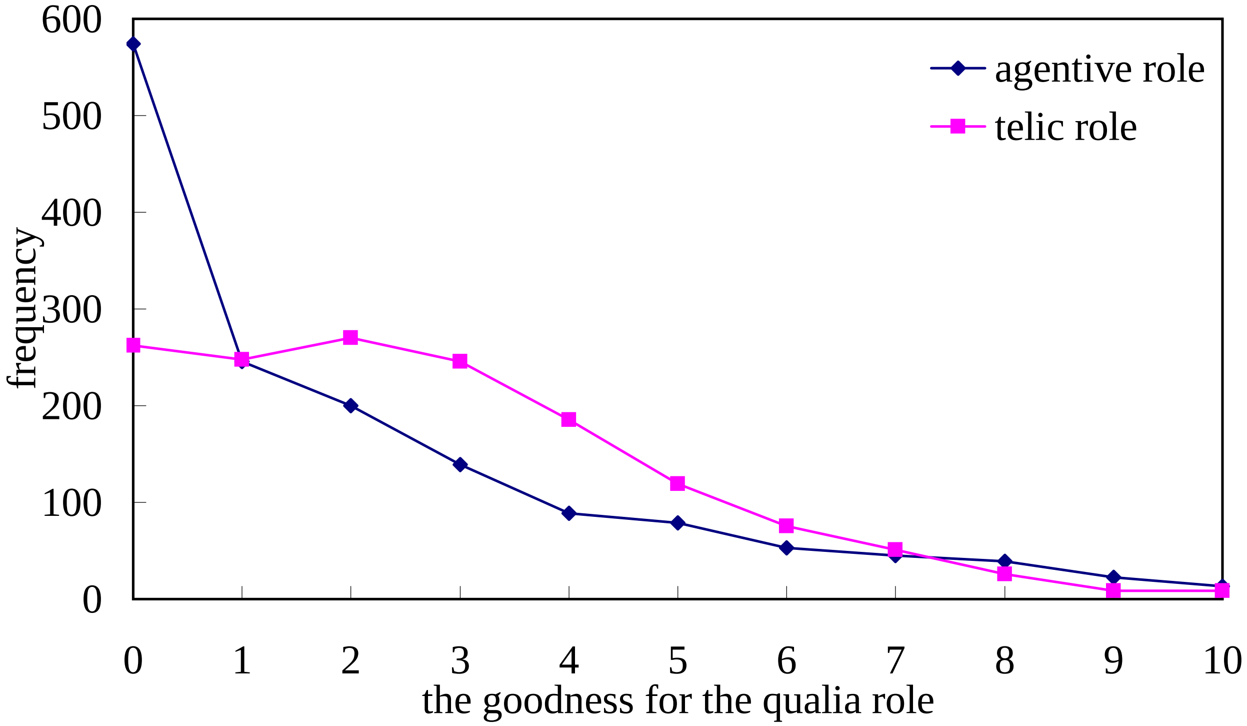
[Oepen et al.2002] Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christoper D. Manning. 2002. LinGO Redwoods: A rich and dynamic treebank for HPSG. In *Proc. of The First Workshop on Treebanks and Linguistic Theories (TLT2002)*, Sozopol, Bulgaria.

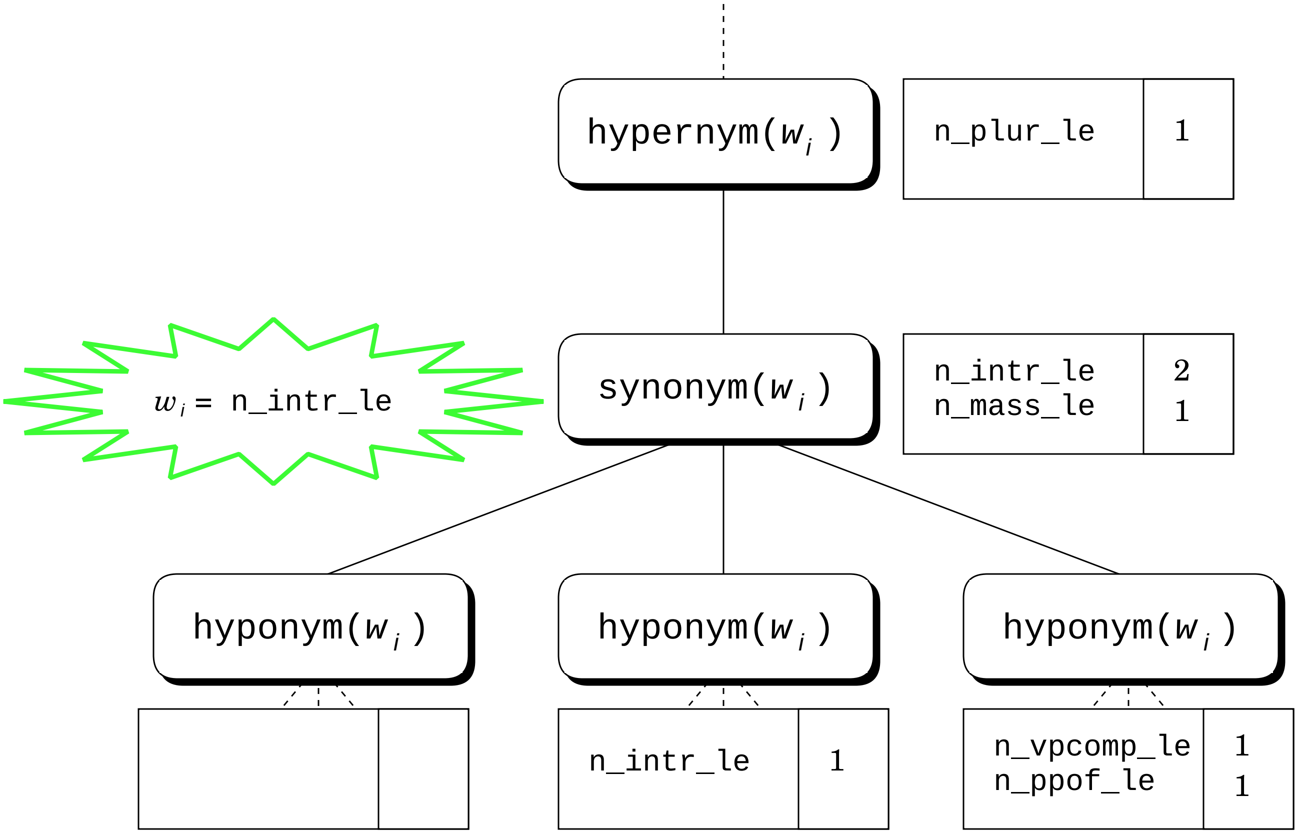
[Sanfilippo and Poznański1992] Antonio Sanfilippo and Victor Poznański. 1992. The acquisition of lexical knowledge from combined machine-readable dictionary sources. In *Proc. of the 3rd Conference on Applied Natural Language Processing (ANLP)*, pages 80–7, Trento, Italy.

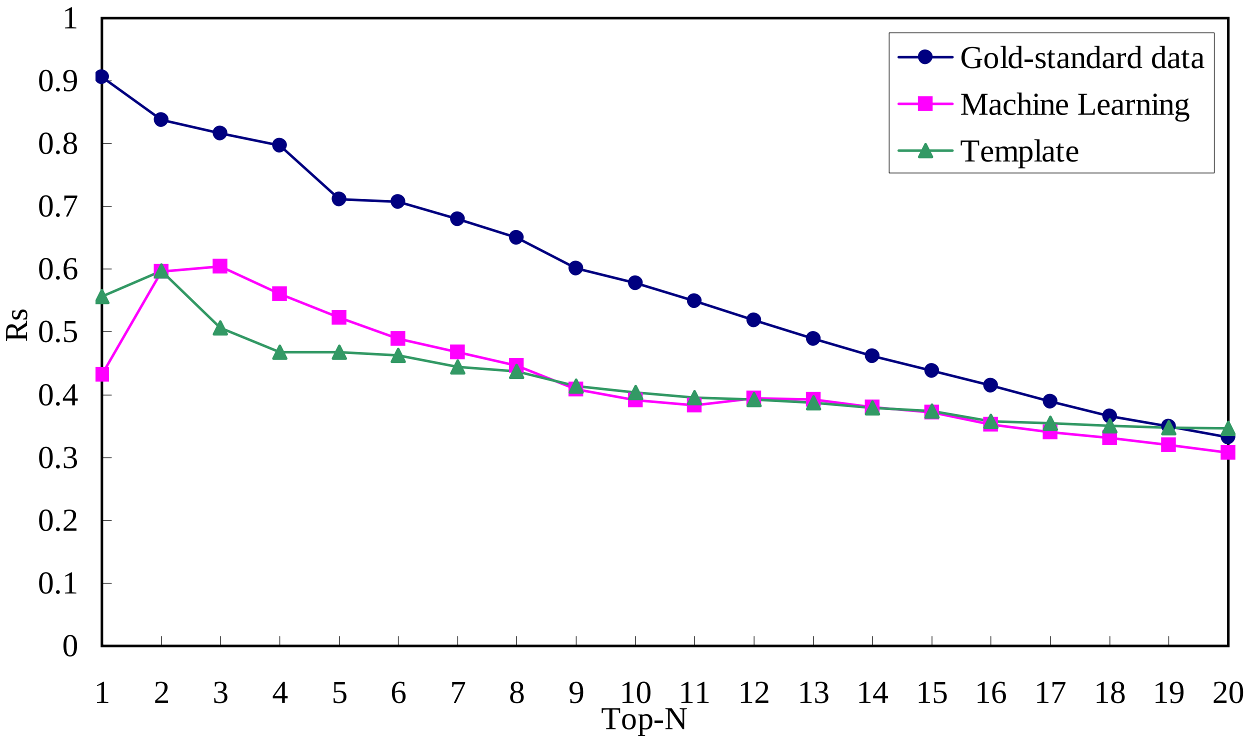
[Tjong Kim Sang and Buchholz2000] Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. of the 4th Conference* *on Computational Natural Language Learning (CoNLL-2000)*, Lisbon, Portugal.

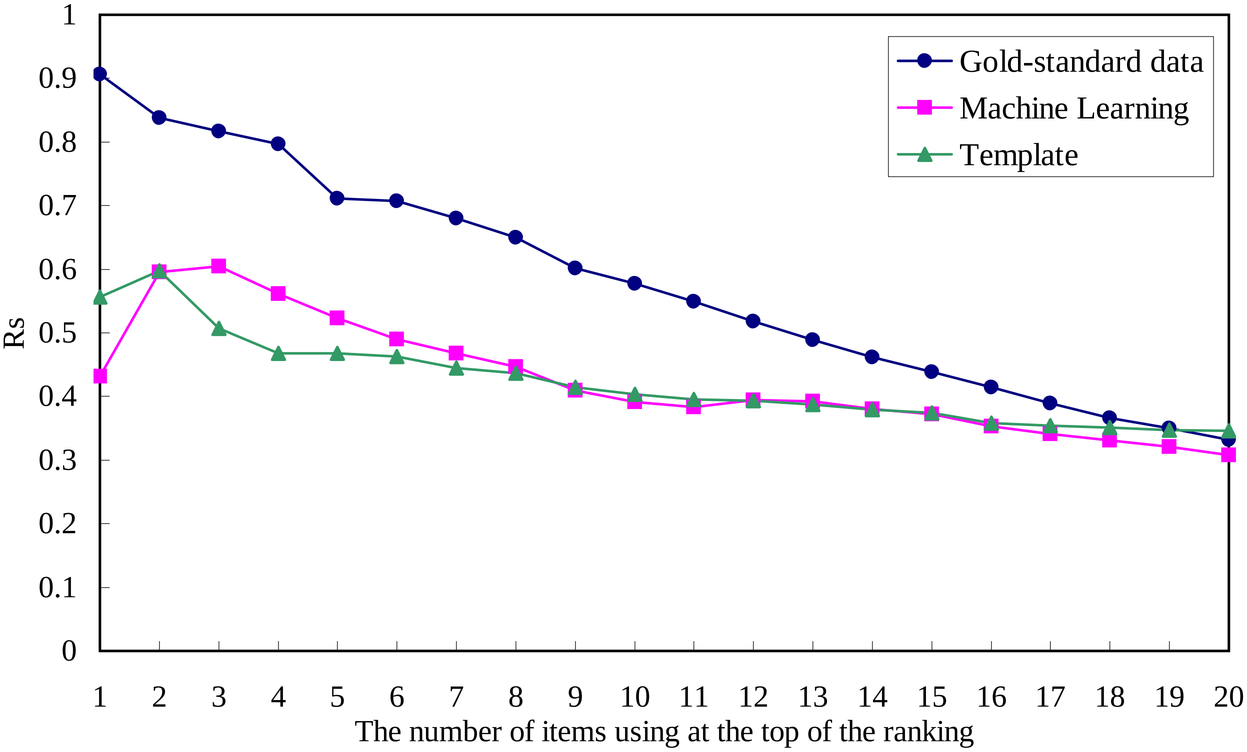
[van der Beek and Baldwin2004] Leonoor van der Beek and Timothy Baldwin. 2004. Crosslingual countability classification with EuroWordNet. In *Papers from the 14th Meeting of Computational Linguistics in the Netherlands*, pages 141–55, Antwerp, Belgium. Antwerp Papers in Linguistics.

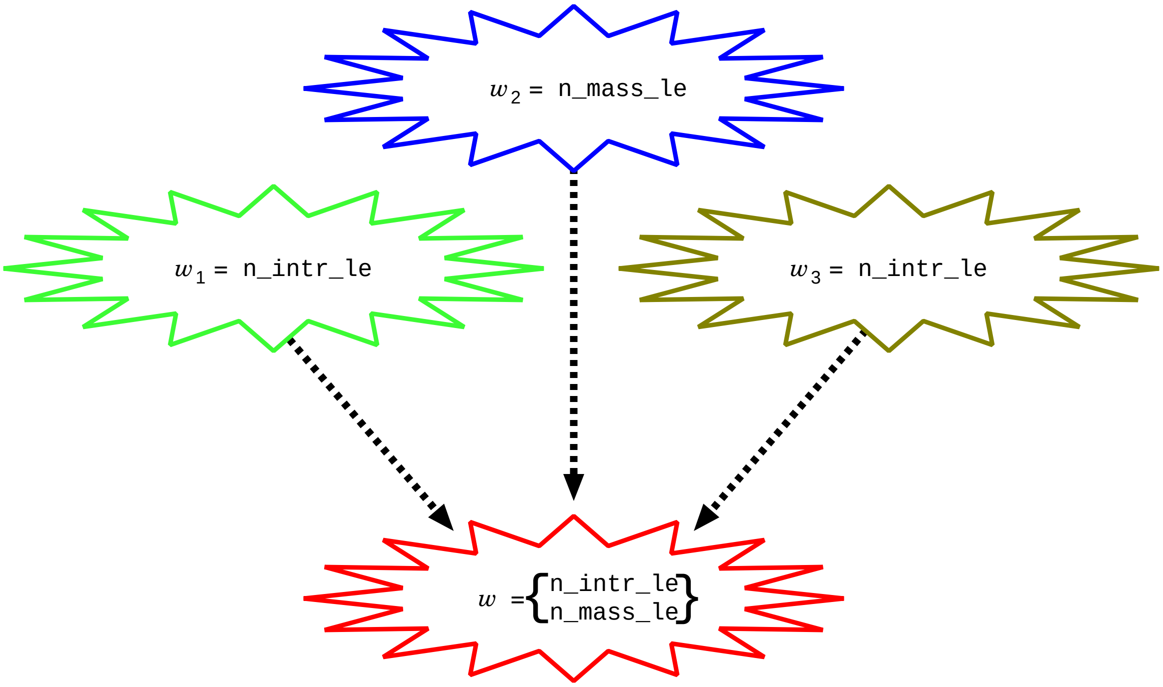

600
500
400
300
200
100
0
frequency
agentive role
telic role
0
1
2
3
4
5
6
7
8
9
10
the goodness for the qualia role

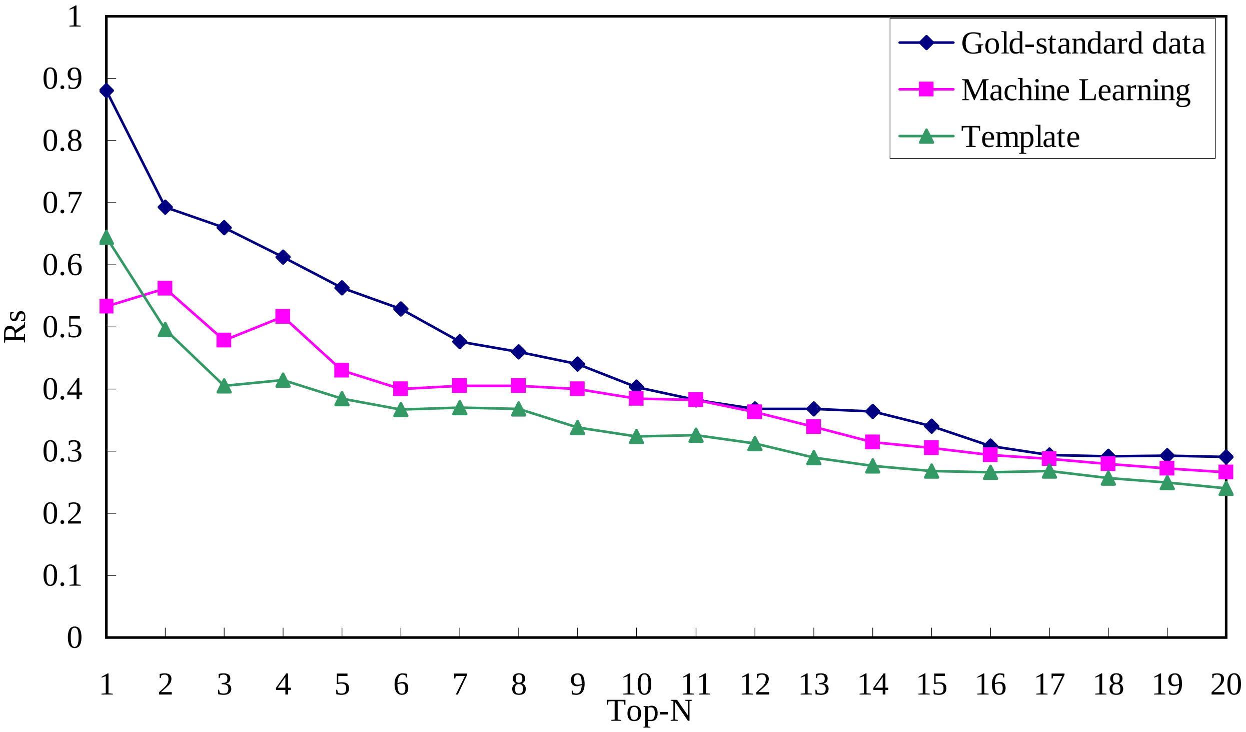

600
500
400
300
200
100
0
frequency
agentive role
telic role
0
1
2
3
4
5
6
7
8
9
10
the goodness for the qualia role

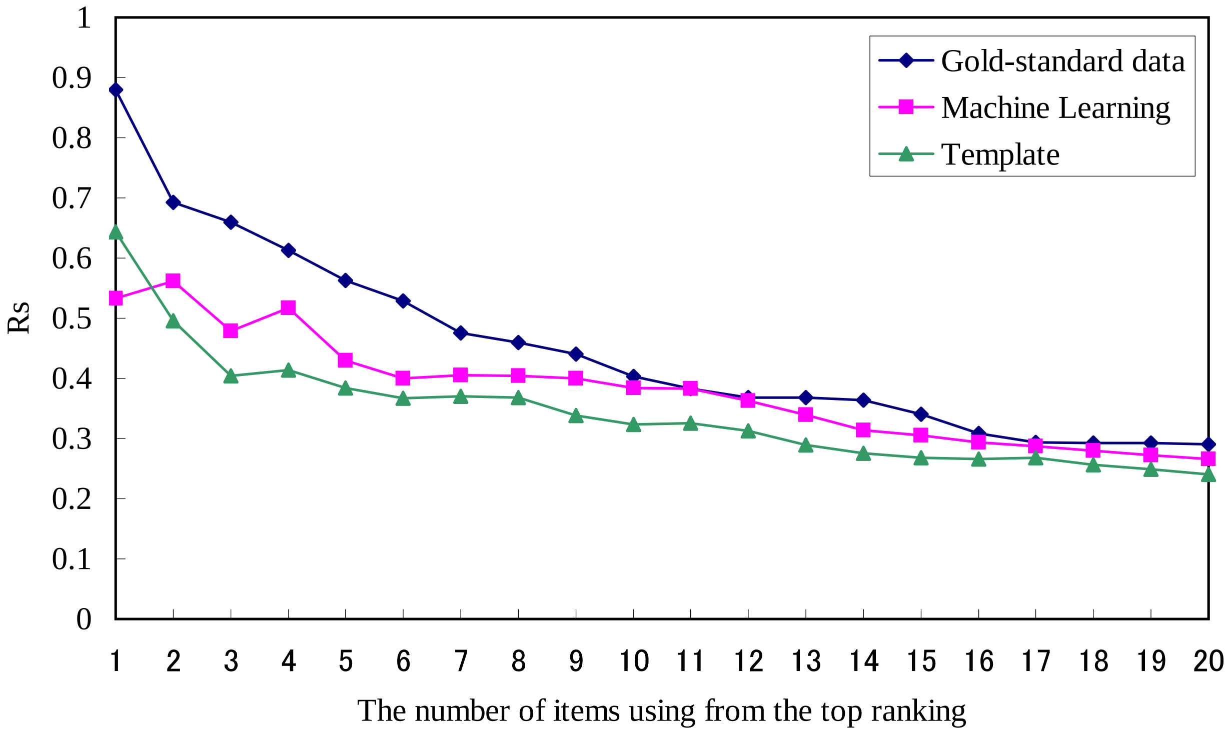

hypernym($w_i$)
n_plur_le
1
$w_i$ = n_intr_le
synonym($w_i$)
n_intr_le
n_mass_le
2
1
hyponym($w_i$)
hyponym($w_i$)
n_intr_le
1
hyponym($w_i$)
n_vpcomp_le
n_ppof_le
1
1


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Rs
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Top-N
Gold-standard data
Machine Learning
Template


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Rs
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
The number of items using at the top of the ranking
Gold-standard data
Machine Learning
Template


$w_2$ = n_mass_le
$w_1$ = n_intr_le
$w_3$ = n_intr_le
$w$ = {n_intr_le
n_mass_le}


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Rs
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Top-N
Gold-standard data
Machine Learning
Template


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Rs
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
The number of items using from the top ranking
Gold-standard data
Machine Learning
Template